



GMOD Chado: to a Model-View-Controller (MVC) architecture?



Valentin GUIGNON
ID, DAP, BIOS
CIRAD Montpellier

Summary

1. Introduction: Current Issues

2. Solutions

2.1 MVC Architecture

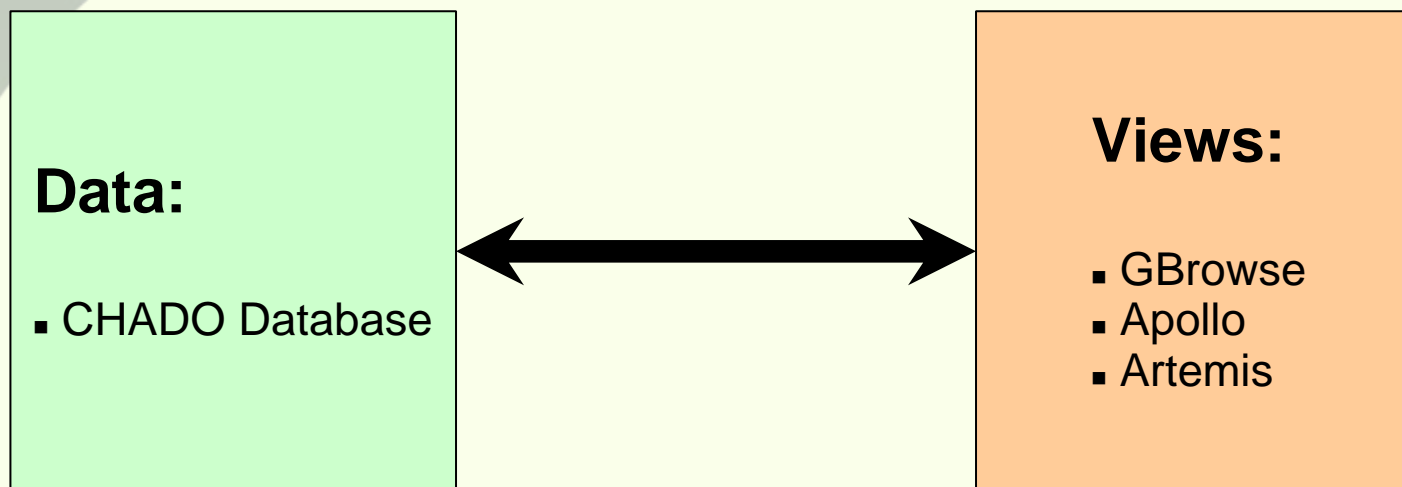
2.2 Integration to GMOD

3. Conclusion



1. Introduction: Current Issues

Current Architecture:



Clients have direct access to the data!



1. Introduction: Current Issues

Issues:

- Access rights (DB): R or R/W
- Edition history
- Concurrent access
- No client compatibility check
- Network security



Summary

1. Introduction: Current Issues

2. Solutions

2.1 MVC Architecture

2.2 Integration to GMOD

3. Conclusion



2.1 Solutions: MVC Architecture

MVC: Model-View-Controller (T.Reenskaug, 1979)

Model: the data stored + the read/write methods

= PostgreSQL or MySQL (CHADO part)

View: the user interface

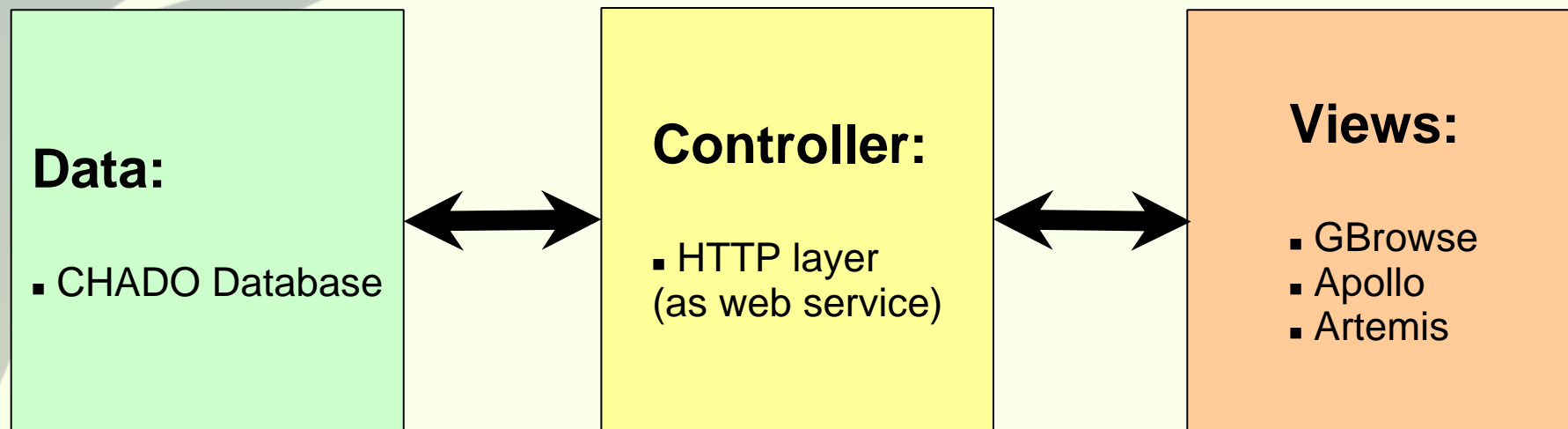
= GBrowse, Apollo, Artemis,...

Controller: handles user actions → effects

= missing!



2.2 Solutions: Integration to GMOD



2.2 Solutions: Integration to GMOD

Tasks:

- Client compatibility check
- User authentication
- Access rights check (queries control)
- History record
- “Data integrity checker” / “annotation inspector”



2.2 Solutions: Integration to GMOD

How will it be achieved?

- Client compatibility check

External DB access will be closed

Protocol will include version check



2.2 Solutions: Integration to GMOD

How will it be achieved?

- User Authentication

Modular: an external module will authenticate users



2.2 Solutions: Integration to GMOD

How will it be achieved?

- Access Rights Check

SQL Queries will be parsed and modified

Some queries may be denied



2.2 Solutions: Integration to GMOD

How will it be achieved?

- History Record

Based on CHADO Audit “module”



Summary

1. Introduction: Current Issues

2. Solutions

2.1 MVC Architecture

2.2 Integration to GMOD

3. Conclusion



3. Conclusion

- A new communication protocol must be set up: wrapped using HTTP(S), similar to SQL syntax
- Requires client side-modification (Apollo, Artemis, GBrowse ...): probably a kind of Java DB driver just like the current ones
- This solution should be submitted to the GMOD community



User Database

annotator

```
(
  id
    integer [PK]
  name
    varchar(32)
  salt
    binary(32)
  password
    binary(32)
  time_registration    timestamp
  time_last_login     timestamp
  time_last_try        timestamp
  failed_login_count   tinyint
```

```
flags
  integer
```

user_group_link

```
(
  group_id integer [FK]
  user_id  integer [FK]
);
```

user_session

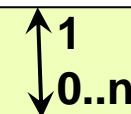
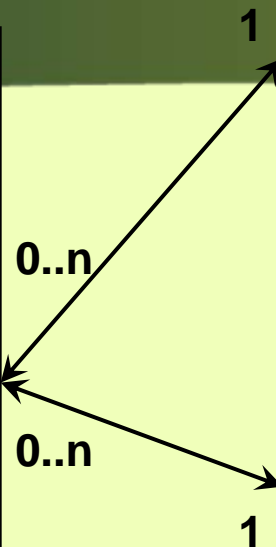
```
(
  session_key    integer [PK]
  user_id        integer
  ip
    binary(16)
  time_expiration timestamp
  access_query   text
```

annotator_feature_access

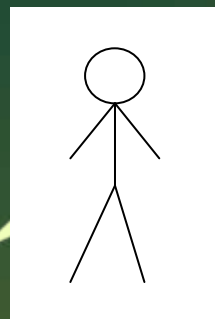
```
(
  feature_id      integer [FK]
  annotator_id    integer [FK]
  access_level    integer
  comment         text
);
```

Feature (from Chado DB)

```
(
  feature_id      integer [PK]
  ...
);
```



Authentication Process



view

Login + Password → HTTPS

OK: Session ID + Groups
Failed: Error code + message

GMOD Controller

OK: Session ID + Groups
Failed: Error code + message

Login + Password
→ Module Call

Authentication
Module

User data

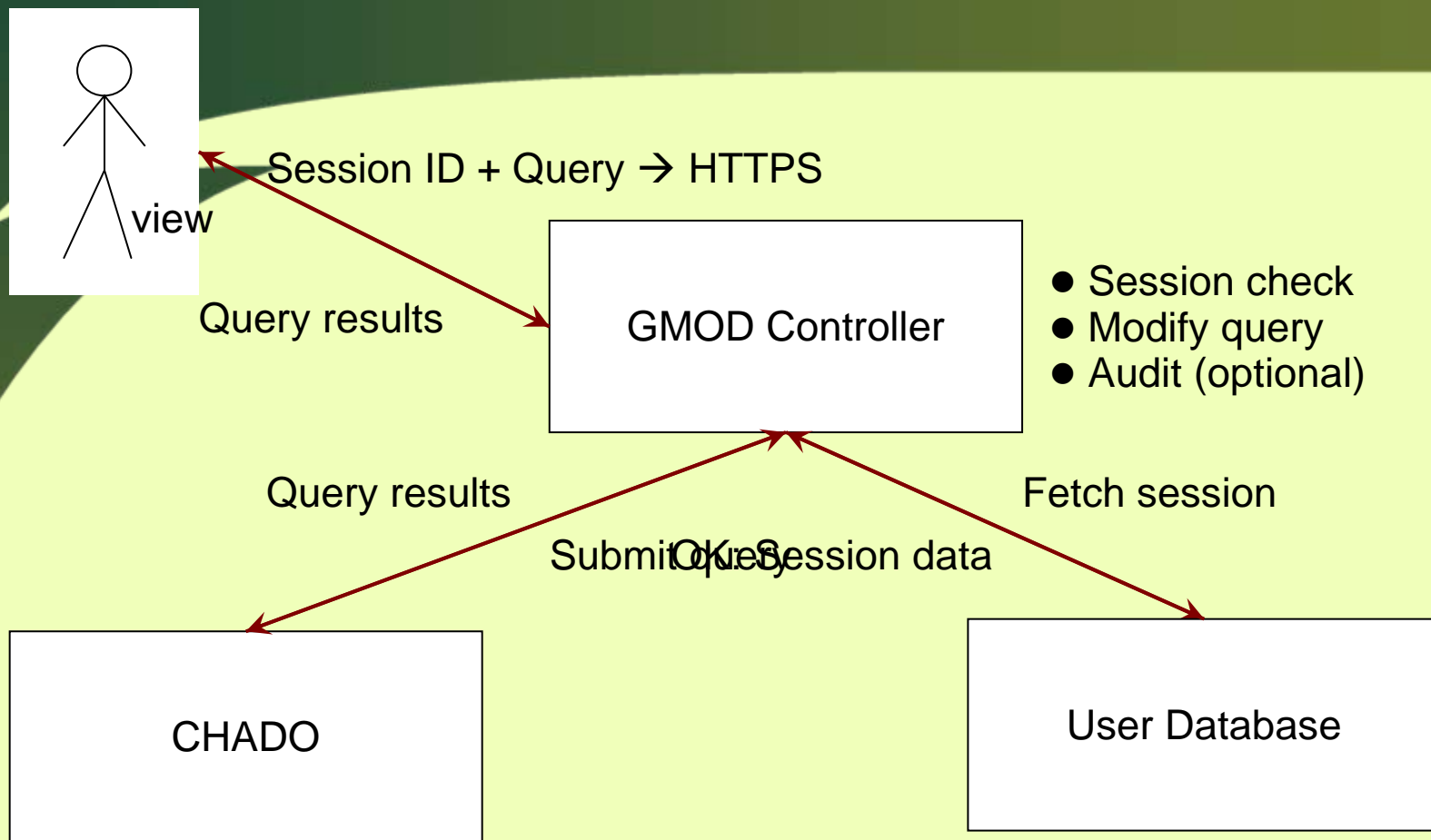
Query user data → SQL

User Database

- Account status check
- Login failures check
- Password check
- Retrieve groups
- Create session



Query Process



Source query:

```
SELECT ... FROM ... feature f
WHERE ... ..;
```

Stored in ***user_session*** table
access_query (set at session
creation time)

Modified Query:

```
SELECT ... FROM ... feature f
WHERE ...
AND f.feature_id IN (
    SELECT fa.feature_id FROM feature_access fa
    WHERE fa.annotator_id IN (
        'UserID', 'Group1ID', 'Group2ID', ...
    )
    AND fa.access >= 1
) ...;
```

